GPS and BlackBerry Maps

Version: 4.6.1

Development Guide

BlackBerry

SWD-596846-1209060923-001

Contents

1 GPS location information ... 3

Specifying the source of GPS location information ... 3

Get location information from cell site towers ... 3

Get location information from GPS satellites using PDE ... 4

Get location information from GPS satellites and the GPS receiver on a BlackBerry device ... 4

Get location information from the GPS receiver on a BlackBerry device ... 5

Getting GPS location information ... 6

Specify a response time for getting the location of the BlackBerry device ... 6

Get the location information for a BlackBerry device ... 6

Get the speed of the BlackBerry device, in meters per second ... 6

Get the course of the BlackBerry device, in degrees ... 7

Get the number of GPS satellites that a BlackBerry device tracks ... 7

Get the time that a BlackBerry device application retrieved GPS information ... 7

Maintain a connection with GPS satellites ... 7

Get continuous GPS readings using the cell site mode ... 7

Get GPS information during a call ... 7

Get location information updates ... 8

Change the criteria for obtaining location information ... 8

2 BlackBerry Maps location information ... 9

Displaying location information on a BlackBerry device ... 9

Displaying location information in BlackBerry Maps ... 9

Open BlackBerry Maps to display the default map view ... 12

Open BlackBerry Maps to display a location on a map ... 13

Open BlackBerry Maps to display multiple locations on a map ... 14

Open BlackBerry Maps to display a route between locations on a map ... 15

Open BlackBerry Maps to display a custom map view ... 17

Open BlackBerry Maps to display the location for an address in the contact list ... 17

Start a BlackBerry device application from BlackBerry Maps ... 19

Displaying location information in your application ... 23

Work with a map that displays in a UI field ... 23

Converting latitudinal and longitudinal coordinates to pixel values ... 24

Requesting location information for an address ... 25

Retrieve location information for an address stored in address fields ... 25

Retrieve location information for an address stored in a String 26
Cancel a request for location information for an address 26
Determine why a request for location information was unsuccessful 26
Open BlackBerry Maps from the browser 26
Clearing data from the map 27
Clear data from a location document with an id attribute 27
Clear data from all location documents with an id attribute 27
Clear data from specific location documents with an id attribute 28
Clear all location data from a map 28

3 Glossary 29

4 Provide feedback 30

5 Legal notice 31

GPS location information

1

You can create a BlackBerry® device application to display the current latitudinal and longitudinal position of a BlackBerry device. Depending on the mode for getting location information that you specify, the BlackBerry device application can also display information such as the speed and route information. To create a BlackBerry device application to get location information, you can use the `Location` API for Java® ME in the `javax.microedition.location` package (JSR 179).

The time it takes to retrieve the location of the BlackBerry device for the first time depends on several factors, such as the mode for getting location information and the GPS signal strength. In autonomous mode, typical times are less than 2 minutes. In assisted mode, typical times are less than 30 seconds.

To retrieve and display GPS location information, the BlackBerry device must support GPS technology and a valid SIM card must be inserted in the BlackBerry device.

Specifying the source of GPS location information

GPS location information might be unavailable if the BlackBerry® device cannot receive signals from GPS satellites. This situation can occur when something is obstructing the signals, such as buildings or dense clouds.

To specify the source of GPS location information, you specify the mode by creating an instance of the `javax.microedition.location.Criteria` class, invoke the appropriate set methods, and then pass the instance to `LocationProvider.getInstance()`.

- cell site: Use this mode to get location information from cell site towers. This mode allows a BlackBerry device application to retrieve location information faster than the assisted and autonomous modes. However, the accuracy of the location information is low-level and does not provide tracking information such as speed or route information. Using this mode requires wireless network coverage and that both the BlackBerry device and the wireless service provider support this mode.
- assisted: Use this mode to get location information from satellites using a PDE. This mode allows a BlackBerry device application to retrieve location information faster than the autonomous mode and more accurately than the cell site mode. To use this mode requires wireless network coverage, and the BlackBerry device and the wireless service provider must support this mode.
- autonomous: Use this mode to get location information from the GPS receiver on the BlackBerry device without assistance from the wireless network. This mode allows a BlackBerry device application to retrieve location information that has high-accuracy, and does not require assistance from the wireless network. However, the speed at which this mode retrieves location information is slower than the other modes.

Get location information from cell site towers

1. Create an instance of a `Criteria` object.

```
Criteria criteria = new Criteria();
```

2. Invoke `criteria.setHorizontalAccuracy(NO_REQUIREMENT)` to specify longitudinal accuracy is not required.
3. Invoke `criteria.setVerticalAccuracy(NO_REQUIREMENT)` to specify latitudinal accuracy is not required.
4. Invoke `criteria.setCostAllowed(true)` to specify that this mode can incur cost.
5. Invoke `setPreferredPowerConsumption(POWER_USAGE_LOW)` to specify power consumption is low.
6. Invoke `LocationProvider.getInstance()`, storing the returned object in a `LocationProvider` object.
```
LocationProvider provider = LocationProvider.getInstance(criteria);
```

Get location information from GPS satellites using PDE

1. Create an instance of a `Criteria` object.
```
Criteria criteria = new Criteria();
```
2. Invoke `criteria.setHorizontalAccuracy(NO_REQUIREMENT)` to specify longitudinal accuracy is not required.
3. Invoke `criteria.setVerticalAccuracy(NO_REQUIREMENT)` to specify latitudinal accuracy is not required.
4. Invoke `criteria.setCostAllowed(true)` to specify that this mode can incur cost.
5. Invoke `criteria.setPreferredPowerConsumption(int level)` using one of `POWER_USAGE_MEDIUM`, or `NO_REQUIREMENT` as the parameter to specify power consumption is medium or not required.
6. Invoke `LocationProvider.getInstance()`, storing the returned object in a `LocationProvider` object.
```
LocationProvider provider = LocationProvider.getInstance(criteria);
```

Get location information from GPS satellites and the GPS receiver on a BlackBerry device

1. Create an instance of a `Criteria` object.
```
Criteria criteria = new Criteria();
```
2. Complete one of the following tasks:

Task	Steps
Specify a level of accuracy for longitudinal and latitudinal data.	a. Invoke `criteria.setHorizontalAccuracy(int accuracy)` using an integer value that represents the accuracy, in meters. b. Invoke `criteria.setVerticalAccuracy(int accuracy)` using an integer value that represents the accuracy, in meters.

Task	Steps
Specify that no level of accuracy for longitudinal and latitudinal data is required,	a. Invoke `criteria.setHorizontalAccuracy(NO_REQUIREMENT).` b. Invoke `criteria.setVerticalAccuracy(NO_REQUIREMENT).`

3. Invoke `criteria.setCostAllowed(true)` to specify that this mode can incur cost.
4. Invoke `criteria.setPreferredPowerConsumption(POWER_USAGE_HIGH)` to specify power consumption is high.
5. Invoke `LocationProvider.getInstance()`, storing the returned object in a `LocationProvider` object.

```
LocationProvider provider = LocationProvider.getInstance(criteria);
```

Get location information from the GPS receiver on a BlackBerry device

1. Create an instance of a `Criteria` object.

```
Criteria criteria = new Criteria();
```

2. Complete one of the following tasks to get location information using the GPS receiver on a BlackBerry device :

Task	Steps
Specify a level of accuracy, with no cost and no power consumption.	a. Invoke `criteria.setHorizontalAccuracy(int accuracy)` using an integer value that represents the accuracy, in meters. b. Invoke `criteria.setVerticalAccuracy(int accuracy)` using an integer value that represents the accuracy, in meters. c. Invoke `criteria.setCostAllowed(false).`
Specify a level of accuracy, with cost and variable power consumption.	a. Invoke `criteria.setHorizontalAccuracy(int accuracy)` using an integer value that represents the accuracy, in meters. b. Invoke `criteria.setVerticalAccuracy(int accuracy)` using an integer value that represents the accuracy, in meters. c. Invoke `criteria.setCostAllowed(true).` d. Invoke `criteria.setPreferredPowerConsumption(int level)` using one of `POWER_USAGE_LOW`, `POWER_USAGE_MEDIUM`,or `NO_REQUIREMENT` as the parameter.

Task	Steps
Specify a level of accuracy, with no cost and variable power consumption.	a. Invoke `criteria.setHorizontalAccuracy(NO_REQUIREMENT)`. b. Invoke `criteria.setVerticalAccuracy(NO_REQUIREMENT)`. c. Invoke `criteria.setCostAllowed(false)`. d. Invoke `criteria.setPreferredPowerConsumption(int level)` using one of `POWER_USAGE_MEDIUM`, `POWER_USAGE_HIGH`, or `NO_REQUIREMENT` as the parameter.

3. Invoke `LocationProvider.getInstance()`, storing the returned object in a `LocationProvider` object.

```
LocationProvider provider = LocationProvider.getInstance(criteria);
```

Getting GPS location information

Specify a response time for getting the location of the BlackBerry device

Invoke `Criteria.setPreferredResponseTime()`, and specify the response time, in milliseconds.

Get the location information for a BlackBerry device

In a non-event thread, invoke `LocationProvider.getLocation(int)`, providing a timeout, in seconds.

```
try {
// Specify -1 to have the implementation use its default timeout value
// for this provider.
Location location = provider.getLocation(-1);
} catch (Exception e) {
// handle LocationException, InterruptedException, SecurityException
// and IllegalArgumentException
}
```

Get the speed of the BlackBerry device, in meters per second

Invoke `location.getSpeed()`.

```
Location location = provider.getLocation(-1);
float speed = mylocation.getSpeed();
```

Get the course of the BlackBerry device, in degrees

Invoke `location.getCourse()`.

```
Location location = provider.getLocation(-1);
float course = mylocation.getCourse();
```

Get the number of GPS satellites that a BlackBerry device tracks

1. Invoke the `Location.getExtraInfo()` method using the NMEA MIME type as a parameter. The method returns an encoded `String` that contains information on the number of GPS satellites.

```
Location location = provider.getLocation(-1);
String encodedStr = mylocation.getExtraInfo("application/X-jsr179-
location-nmea");
```

2. To retrieve the GPS satellite information, parse through the information in the returned `String` value.

Get the time that a BlackBerry device application retrieved GPS information

Invoke `Location.getTimeStamp()`. The time is retrieved from the GPS receiver on the BlackBerry® device.

```
Location location = provider.getLocation(-1);
myLocation.getTimeStamp();
```

Maintain a connection with GPS satellites

You can configure a BlackBerry® device application to query the Location API for GPS location information in an interval less than 10 seconds.

Define a `LocationListener` with an interval value that is less than 10 seconds.

```
LocationProvider provider = LocationProvider.getInstance(criteria);
provider.setLocationListener(loclistener, 9, -1, -1);
```

Get continuous GPS readings using the cell site mode

Invoke the `LocationProvider.getLocation()` method within code that simulates the retrieval of GPS information at fixed intervals.

Get GPS information during a call

Use the autonomous mode to retrieve GPS information.

Get location information updates

You can associate only one `LocationListener` object with a particular provider for GPS location information. The BlackBerry® device application typically listens for updates on a separate thread.

1. Implement the `LocationListener` interface.
2. To register your implementation, invoke `LocationProvider.setLocationListener()`.

Change the criteria for obtaining location information

1. Create a new `Criteria` object.
   ```
   Criteria criteria2 = new Criteria();
   ```
2. Invoke the `setHorizontalAccuracy()`, `setVerticalAccuracy()`, `setCostAllowed()`, and `setPreferredPowerConsumption()` methods of the new `Criteria` object to specify the new criteria.
3. Invoke `LocationProvider.reset()`.
   ```
   LocationProvider.reset();
   ```
4. Specify the `LocationListener` to null.
   ```
   provider.setLocationListener(null, -1, -1, -1);
   ```
5. Invoke `LocationProvider.getInstance()` using the new `Criteria` object as a parameter to create a new instance of a `LocationProvider`.
   ```
   provider = LocationProvider.getInstance(criteria2);
   ```
6. Specify the `LocationListener`.
   ```
   provider.setLocationListener(LocationListener listener, int interval, int timeout, int maxAge);
   ```

BlackBerry Maps location information

2

You can create a BlackBerry® device application that interacts with BlackBerry® Maps. The BlackBerry Maps is a map and location client application that can display a map for a location, the location of the BlackBerry device, a route from a starting location to a specific ending location, and points of interest on a map. The BlackBerry Maps uses latitudinal and longitudinal integer values that are 100,000 times the double values specified by JSR 179, which uses WGS 84 specifications for the longitudinal and latitudinal values.

A BlackBerry device application can interact with the BlackBerry Maps in the following ways:

- start the BlackBerry Maps from a BlackBerry device application
- add a menu item to the BlackBerry Maps that opens a BlackBerry device application

The BlackBerry Maps is included on BlackBerry devices that run BlackBerry® Device Software version 4.2 or later.

Displaying location information on a BlackBerry device

Displaying location information in BlackBerry Maps

The BlackBerry® Maps uses a location document to display location information on a map, such as locations and routes. A location document is a `String` that contains a set of XML elements with attributes that specify information about the location or route. You can include information for ten locations in one location document.

XML element: <lbs>

The `<lbs>` and `</lbs>` elements encapsulate the information in a location document. The opening and closing `<lbs>` elements contain all other location document elements.

Attribute	Type	Description	Required	BlackBerry Java Development Environment version
`id`	`String`	the id of a location document	no	4.5.0 or later
`clear`	`String`	the action to perform on the information in a map NONE: clears no information	no	4.5.0 or later

Attribute	Type	Description	Required	BlackBerry Java Development Environment version
		DOCS: clears location or route information from all location documents with a id attribute set to a value LOCATIONS: clears location information from the map ALL: clears location and route information from the map		

XML element: <getRoute>

The `<getRoute>` and `</getRoute>` elements contain route information. To display route information in a map, you must place two `<location>` elements within the opening and closing `<getRoute>` elements. The first `<location>` element represents the starting point of a route while the second `<location>` element represents the ending point of a route. You can only use the x and y attributes in a `<location>` element that is nested within a `<getRoute>` element.

XML element: <location>

The `<location>` element contains information for a specific location and has the following required and optional attributes

Attribute	Type	Description	Required	BlackBerry Java Development Environment version
y	`integer`	latitude in decimal degrees x 100,000	yes	4.2.0 or later
x	`integer`	longitude in decimal degrees x 100,000	yes	4.2.0 or later
zoom	`integer`	zoom level from 0 to MAX_ZOOM	yes	4.2.0 or later

Attribute	Type	Description	Required	BlackBerry Java Development Environment version
label	`String`	a label that is displayed beside a location on a map	yes	4.2.0 or later
description	`String`	description information for a location	yes	4.2.0 or later
address	`String`	address	no	4.2.1 or later
city	`String`	city	no	4.2.1 or later
region	`String`	province or state	no	4.2.1 or later
country	`String`	country	no	4.2.1 or later
postalCode	`String`	postal code	no	4.2.1 or later
phone	`String`	phone number	no	4.2.1 or later
fax	`String`	fax number	no	4.2.1 or later
url	`String`	URL	no	4.2.1 or later
email	`String`	email address	no	4.2.1 or later
categories	`String`	category	no	4.2.1 or later
rating	`String`	rating information, between 0 and 5	no	4.2.1 or later

Create a location document to display location information

1. Create a `String` using required elements, or required and optional elements.

```
String document = "<lbs>
<location y='latitude' x='longitude' label='Location_Label'
description='Description'/>
<location y='latitude' x='longitude' label='Location_Label'
description='Description'/>
<location y='latitude' x='longitude' label='Location_Label'
description='Description'/>
....</lbs>";
```

2. Invoke `invokeApplication()` using the `String` as one of the parameters.

```
Invoke.invokeApplication(Invoke.APP_TYPE_MAPS, new MapsArguments
( MapsArguments.ARG_LOCATION_DOCUMENT, document));
```

Open BlackBerry Maps to display the default map view

1. Import the following classes:
   - `net.rim.blackberry.api.invoke.Invoke`
   - `net.rim.blackberry.api.invoke.MapsArguments`
2. Invoke `invokeApplication()` using a new `MapsArguments` object that does not contain any arguments.

View a sample application that displays the default map

1. From the BlackBerry® Integrated Development Environment, launch the **BlackBerryMapsDemo** project.
2. In the BlackBerry Smartphone Simulator, click **BlackBerry Maps Demo**.
3. Open the menu and click **Invoke Default**.
4. Open the menu and click **View Map**.

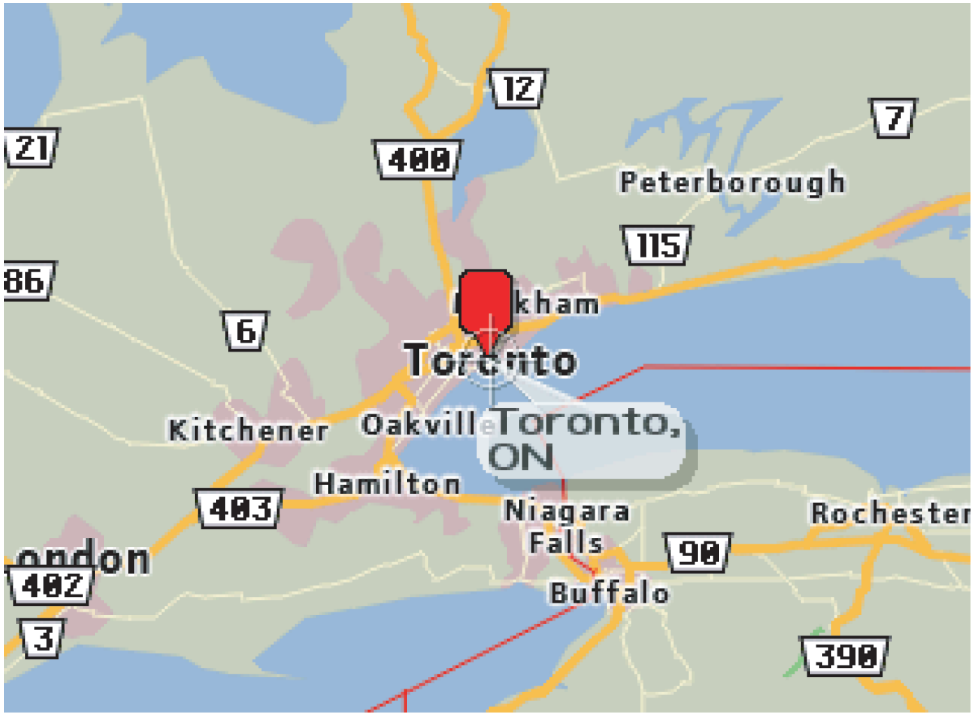
Open BlackBerry Maps to display a location on a map

1. Import the following classes:
   - `net.rim.blackberry.api.invoke.Invoke`
   - `net.rim.blackberry.api.invoke.MapsArguments`
2. Create a `String` that contains location information for one location.

```
String document = "<location-document><location lon='-7938675'
lat='4367022' label='Toronto, ON' description='Toronto' zoom='10'/></
location-document>";
```

3. Invoke `invokeApplication()` using the `APP_TYPE_MAPS` constant parameter, a new `MapsArguments` object that uses the `ARG_LOCATION_DOCUMENT` property, and the location `String`.

```
Invoke.invokeApplication(Invoke.APP_TYPE_MAPS, new MapsArguments
( MapsArguments.ARG_LOCATION_DOCUMENT, document));
```



View a sample application that displays a location on a map

1. From the BlackBerry® Integrated Development Environment, launch the **BlackBerryMapsDemo** project.
2. In the BlackBerry Smartphone Simulator, click **BlackBerry Maps Demo**.
3. Open the menu and click **Invoke Location Document**.
4. Open the menu and click **View Single Location**.
5. To view more details, from the application menu, select **Details**.

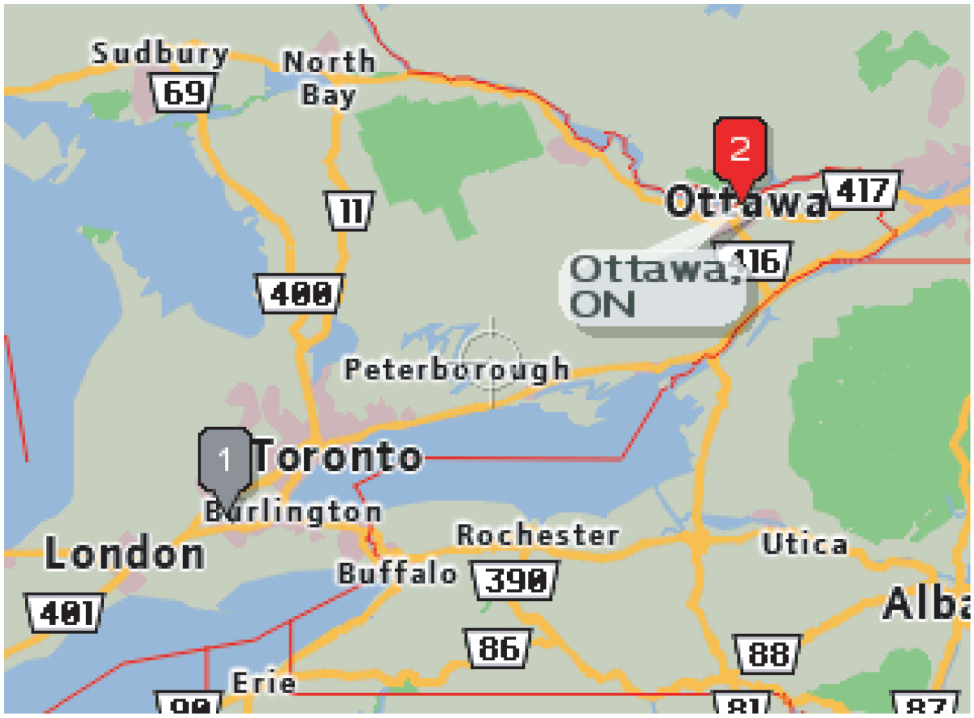
Open BlackBerry Maps to display multiple locations on a map

1. Import the following classes:
   - `net.rim.blackberry.api.invoke.Invoke`
   - `net.rim.blackberry.api.invoke.MapsArguments`
2. Create a `String` that contains location information for multiple locations.

```
String document = "<location-document>" + "<location lon='-8030000'
lat='4326000' label='Kitchener, ON' description='Kitchener, Ontario,
Canada' />" + "<location lon='-7569792' lat='4542349' label='Ottawa, ON'
description='Ottawa, Ontario, Canada' />" + "</location-document>";
```

3. Invoke `invokeApplication()` using the `APP_TYPE_MAPS` constant parameter, a new `MapsArguments` object that uses the `ARG_LOCATION_DOCUMENT` property, and the location `String`.

```
Invoke.invokeApplication(Invoke.APP_TYPE_MAPS, new MapsArguments
( MapsArguments.ARG_LOCATION_DOCUMENT, document));
```



View a sample application that displays multiple locations on a map

1. From the BlackBerry® Integrated Development Environment, launch the **BlackBerryMapsDemo** project.
2. In the BlackBerry Smartphone Simulator, click **BlackBerry Maps Demo**.
3. Open the menu and click **Invoke Location Document**.
4. Open the menu and click **View Multiple Locations**.

Open BlackBerry Maps to display a route between locations on a map

1. Import the following classes:
   - `net.rim.blackberry.api.invoke.Invoke`
   - `net.rim.blackberry.api.invoke.MapsArguments`
2. Create a `String` with route information within the `<getRoute>` elements.

```
String document = "<location-document><GetRoute>" + "<location
lon='-8030000' lat='4326000' label='Kitchener, ON' description='Kitchener,
Ontario, Canada' />" +
```

```
"<location lon='-7569792' lat='4542349' label='Ottawa, ON'
description='Ottawa, Ontario, Canada' />" + "</GetRoute></location-
document>";
```

3. Invoke `invokeApplication()` using the `APP_TYPE_MAPS` constant parameter, a new `MapsArguments` object that uses the `ARG_LOCATION_DOCUMENT` property, and the `String`.

```
Invoke.invokeApplication(Invoke.APP_TYPE_MAPS, new MapsArguments
( MapsArguments.ARG_LOCATION_DOCUMENT,document));
```

View a sample application that displays a route between locations on a map

1. From the BlackBerry® Integrated Development Environment, launch the **BlackBerryMapsDemo** project.
2. In the BlackBerry Smartphone Simulator, click **BlackBerry Maps Demo**.
3. Open the menu and click **Invoke Location Document**.
4. Open the menu and click **View Route**.
5. To view tracking information from one location to another, from the application menu, select **Directions**.

Open BlackBerry Maps to display a custom map view

A `MapView` object represents the view that the BlackBerry® Maps displays. The `MapView` object contains the latitude and longitude that the BlackBerry Maps uses as the center of the map and the zoom and rotation values for the map that it displays.

1. Import the `net.rim.blackberry.api.maps.MapView` class.
```
import net.rim.blackberry.api.maps.MapView;
```
2. Create an instance of a `MapView` object.
```
MapView mapView = new MapView();
```
3. Modify the latitude, longitude, and zoom values for the `MapView` object.
```
mapView.setLatitude(4331168);
mapView.setLongitude(-8055339);
mapView.setZoom(10);
```
4. Invoke `invokeApplication()` using the `APP_TYPE_MAPS` constant parameter, and a new `MapsArguments` object that uses the `MapView` object you created.
```
Invoke.invokeApplication(Invoke.APP_TYPE_MAPS, new MapsArguments
(mapView));
```

View a sample application that displays a custom map view

1. From the BlackBerry® Integrated Development Environment, launch the **BlackBerryMapsDemo** project.
2. In the BlackBerry Smartphone Simulator, click **BlackBerry Maps Demo**.
3. Open the menu and click **Invoke Map View**.
4. Open the menu and click **View Map**.

Open BlackBerry Maps to display the location for an address in the contact list

You can create a BlackBerry® Application that displays a map for the address of a contact in the contact list. The address for the contact must be a valid address.

1. Import the `javax.microedition.pim.PIM` class and the `javax.microedition.pim.Contact` interface.
```
import javax.microedition.pim.PIM;
import javax.microedition.pim.Contact ;
```
2. Create a `Contact` object.
```
Contact c = null;
```
3. In a try block, get the contact list.
```
try {
ContactList contactList = (ContactList)PIM.getInstance().openPIMList
(PIM.CONTACT_LIST, PIM.READ_WRITE);
```

4. Place the items from the contact list into an enumeration.

```
Enumeration enumContact = contactList.items();
int notFinished   = 1;
```

5. Create a `While` loop to iterate through all the items in the enumeration.

```
while ((enumContact.hasMoreElements())  && ( notFinished == 1)) {
c = (Contact)enumContact.nextElement();
int[]   fieldIds = c.getFields();
int     id;
```

6. Search for the first contact with a valid city and region and display the address information for this contact in the BlackBerry® Maps.

```
if (c.countValues(Contact.ADDR) > 0) {
String address[] = c.getStringArray(Contact.ADDR,0);
if ((address[Contact.ADDR_LOCALITY] != null) && (address
[Contact.ADDR_REGION] != null)) {
              Invoke.invokeApplication(Invoke.APP_TYPE_MAPS, new
MapsArguments(c,0));
              notFinished = 0;
            }
        }
     }
} catch (PIMException e) {
    }
```

7. If a contact address is not found, display the default map in the BlackBerry Maps.

```
if  © == null) {
Invoke.invokeApplication(Invoke.APP_TYPE_MAPS, new MapsArguments());
}
}
};
```

View a sample application that displays the location for an address in the contact list

Before you begin:

On a BlackBerry® device, open the contact list. If there are no addresses in the contact list, the sample will display the default view in the BlackBerry® Maps.

1. Open the BlackBerry® Integrated Development Environment.
2. Launch the **BlackBerryMapsDemo** project.
3. In the BlackBerry Smartphone Simulator, click **BlackBerry Maps Demo**.
4. Open the menu and click **Invoke Contact**.
5. Open the menu and click **View Map**.

Start a BlackBerry device application from BlackBerry Maps

You can add a menu item to BlackBerry® Maps that makes a `MapView` context object available based on the current location. When a BlackBerry device user selects the menu item, the `MapView` context object is passed to the menu item's `run()` method. The BlackBerry device user can modify the `MapView` information through the application's UI and view an updated map based on that input.

1. Open the BlackBerry® Integrated Development Environment.
2. Create a project.
3. Right-click the project, and click **Properties**.
4. Click the **Application** tab.
5. In the **Arguments passed to** field, type **startup**. Make sure the value you type matches the value in the **startsWith** argument in the main method of the BlackBerry Application.
6. Select **Auto-run on startup**.
7. Select **System module**.
8. Click **OK**.
9. Import the following classes:

```
import net.rim.device.api.ui.*;
import net.rim.device.api.ui.component.*;
import net.rim.device.api.ui.container.*;
import net.rim.blackberry.api.invoke.*;
import net.rim.device.api.system.*;
import net.rim.blackberry.api.maps.*;
import net.rim.blackberry.api.menuitem.*;
```

10. Create a class that extends the `ApplicationMenuItem` class and implements the `run()` and `toString()` methods.

```
private static class MapsMenuItem extends ApplicationMenuItem
    {
// Constructor
        MapsMenuItem()
        {
            // Create a new ApplicationMenuItem instance with a relative
menu
            // position of 20. Lower numbers correspond to higher
placement
            // in the menu.
            super(20);
        }
        /**
         * Returns the name to display in a menu.
         * @return The name to display.
```

```
         */
public String toString()
        {
            return "Menu Item Demo";
        }
        /**
         * Code that executes when a BlackBerry smartphone user clicks the
menu item.
         * @param context A MapView object.
* @return null
         */
        public Object run(Object context)
        {
            if (context instanceof MapView )
            {
                _mv = (MapView)context;
                //  Obtain the instance of the UiApplication and display
the UI screen.
                UiApplication app = UiApplication.getUiApplication();
app.pushScreen( new MapsMenuItemScreen(_mv) );
                app.requestForeground();
            }
            return null;
        }
    }
}
```

11. In the `main()` method, register a menu item with BlackBerry Maps.

```
public static void main(String[] args) {
if (args != null && args.length > 0)
        {
            if (args[0].equals("startup"))
{
                // Register an ApplicationMenuItem when the BlackBerry
smartphone starts.
                ApplicationMenuItemRepository amir =
ApplicationMenuItemRepository.getInstance();
                ApplicationDescriptor ad_startup =
ApplicationDescriptor.currentApplicationDescriptor();
                ApplicationDescriptor ad_gui = new ApplicationDescriptor
(ad_startup , "gui", new String[]{"gui"});
                amir.addMenuItem
(ApplicationMenuItemRepository.MENUITEM_MAPS , new MapsMenuItem() ,
ad_gui);
            }
            else if (args[0].equals("gui"))
            {
```

```
                // The ApplicationMenuItem invoked App . Call the default
                // constructor for the GUI version of the application and
enter the
application into the main event thread.
                MapsMenuItemDemo app = new MapsMenuItemDemo();
                app.enterEventDispatcher();
            }
        }
    }
```

12. Create a screen for the application.

```
final class MapsMenuItemScreen extends MainScreen
{
    private MapView _mapview;
    private BasicEditField _latitudeField;
    private BasicEditField _logitudeField;
    private NumericChoiceField _zoomField;
    //Constructor
    /**
     * @param _mv The MapView context object.
*/
    MapsMenuItemScreen(MapView _mv)
    {
        _mapview = _mv;
        // The int values that getLatitude() and getLongitude() return are
100,000 times
        // the values specified by WGS84.
        _latitudeField = new BasicEditField ("Latitude:      " ,
_mv.getLatitude()/100000.0 + "" , 9 ,
BasicEditField.FILTER_REAL_NUMERIC );
        _logitudeField = new BasicEditField ("Longitude:   " ,
_mv.getLongitude()/100000.0 + "" , 10 ,
BasicEditField.FILTER_REAL_NUMERIC);
        _zoomField = new NumericChoiceField ("Zoom: " , 0 ,
MapView.MAX_ZOOM , 1 , _mv.getZoom());
        // Add GUI components.
        add(_latitudeField);
        add(_logitudeField);
        add(_zoomField);
        add(new SeparatorField());
        add(new RichTextField("Edit latitude, longitude and zoom level
settings and select View Map from the menu." ,
Field.NON_FOCUSABLE));
        addMenuItem(viewMapItem);
        setTitle("Location Details Screen");
    }
```

13. Create a menu item that displays the latitude, longitude, and zoom values from the `MapView` context object.

```
private MenuItem viewMapItem = new MenuItem("View Map", 1000, 10)
    {
        public void run()
        {
            // Change the zoom level.
_mapview.setZoom( _zoomField.getSelectedValue() );
            try
            {
                int latitude = (int) (100000 * Double.parseDouble
(_latitudeField.getText()));
                int longitude = (int) (100000 * Double.parseDouble
(_logitudeField.getText()));
               if (latitude > 9000000 || latitude < -9000000 || longitude
>= 18000000 || longitude < -18000000)
                {
                    throw new IllegalArgumentException ();
                }
                _mapview.setLatitude(latitude);
                _mapview.setLongitude(longitude);
                // Invoke BlackBerry Maps with a MapView object.
                Invoke.invokeApplication(Invoke.APP_TYPE_MAPS, new
MapsArguments(_mapview));
close();
            }
            catch(RuntimeException re)
            {
                // An exception is thrown when any of the following
occur :
                // Latitude is invalid : Valid range: [-90, 90]
                // Longitude is invalid : Valid range: [-180, 180)
                // Minus sign between 2 numbers.
                Dialog.alert("Invalid Longitude and/or Latitude");
                _latitudeField.setFocus();
            }
        }
    };
```

14. Populate the menu.

```
protected void makeMenu( Menu menu, int instance )
    {
        super.makeMenu( menu, instance );
        // If _zoomField has focus , make "Change Option" item default.
        if(getFieldWithFocus() == _zoomField)
        {
```

```
            menu.setDefault(1);
        }
}
```

View a sample application that starts a BlackBerry Application from BlackBerry Maps

1. From the BlackBerry® Integrated Development Environment, launch the BlackBerry Smartphone Simulator.
2. In the BlackBerry Smartphone Simulator, launchBlackBerry Maps.
3. In BlackBerry Maps, click **Menu Item Demo**.
4. Complete the instructions on the screen and click **View Map**.

Displaying location information in your application

A BlackBerry® device application can use the `MapField` class to display a map in a UI field. For a BlackBerry device application to use the `MapField` class, BlackBerry® Maps must be installed on the BlackBerry device and the BlackBerry device must be able to download data from the map server for location-based services.

A `MapField` gets data either from a map server for location-based services or the BlackBerry device with BlackBerry Maps installed. If location information is not present on the BlackBerry device, the `MapField` requests data from the map server. See the API reference for the BlackBerry Java Development Environment for more information about the `MapField` API.

Work with a map that displays in a UI field

1. Import the `net.rim.device.api.lbs.MapField` class.
2. Invoke any of the following methods:

Task	Steps
Modify the center of a `MapField` using latitudinal and longitudinal values, in degrees.	Invoke `moveTo(int latitude, int longitude)`. The latitude and longitude values are calculated as degrees multiplied by 100,000.
Modify the center of a `MapField` to specific coordinates.	Invoke `moveTo(javax.microedition.location.Coordinates coords)`.

Task	Steps
Modify the center of a `MapField` using pixel values.	Invoke `move(int dx,int dy)`. The `move(dx,dy)` method moves center of the map in terms of relative pixels. A positive `dx` moves the center of the map east if the rotation is 0, and a positive `dy` moves the center of the map north if the rotation is 0.
Modify the zoom value for a `MapField`.	Invoke `setZoom(int z)`. Zoom values can be from 0 to 15. The zoom operation is performed to the center of the map in the `MapField`.
Modify the rotation for a `MapField`, in degrees.	Invoke `setRotation(int r)`. Rotation can be set from 0 to 360, clockwise. A rotation setting of 0 sets north at the top of the map view.
Retrieve the latitude of the center of a `MapField`.	Invoke `getLatitude()`.
Retrieve the longitude of the center of a `MapField`.	Invoke `getLongitude()`.
Retrieve the zoom value of a `MapField`.	Invoke `getZoom()`.
Retrieve the rotation setting for a `MapField`.	Invoke `getRotation()`.

Converting latitudinal and longitudinal coordinates to pixel values

You can create a BlackBerry® device application to convert latitudinal and longitudinal coordinates to pixel values. For latitude, north is a positive value, and south is a negative value. For longitude, east is a positive value and west is a negative value.

The `Coordinates` class represents coordinates as latitudinal, longitudinal, and altitudinal values. The latitudinal and longitudinal values are expressed in degrees using floating point values. The `Coordinates` class provides coordinates using World Geodetic System 1984 (WGS 84) data.

Convert latitudinal and longitudinal coordinates

The `XYPoint` object stores the initial coordinates and changes to the converted coordinates.

1. Import the `net.rim.device.api.lbs.MapField` class.
2. To convert latitudinal and longitudinal coordinates to pixels, invoke `convertWorldToField(Coordinates worldIn, XYPoint fieldOut)` or `convertWorldToField(XYPoint worldIn, XYPoint fieldOut)`.

3. To convert pixels to latitudinal and longitudinal coordinates, invoke `convertFieldToWorld(XYPoint fieldIn, Coordinates worldOut)` or `convertFieldToWorld(XYPoint fieldIn, XYPoint worldOut)`.

Configure the display size of a MapField

A BlackBerry® device application can control the display size of a `MapField` to make other UI components visible on the screen of a BlackBerry device.

If you configure the preferred size of `MapField`, you override the dimension values passed in `layout()`. The height and width values are in pixels.

Invoke `setPreferredSize(int preferredWidth, int preferredHeight)`.

Requesting location information for an address

You can use the `Locator` class in a BlackBerry® device application to request location information for an address. Requests are sent to a locator server for location-based services. If the request is successful, the server returns an enumeration that contains latitudinal and longitudinal data for the address. If the request is unsuccessful, the server throws a`LocatorException` exception that contains the reason for the unsuccessful request. A BlackBerry device application can request location information for only one address at a time.

Retrieve location information for an address stored in address fields

1. Create an `AddressInfo` object.
```
AddressInfo ai = new AddressInfo();
```
2. Set the fields of the `AddressInfo` object.
```
ai.setField(AddressInfo.STREET, "main street");
ai.setField(AddressInfo.CITY, "Toronto");
ai.setField(AddressInfo.STATE, "Ontario");
ai.setField(AddressInfo.POSTAL_CODE, "XXX XXX");
ai.setField(AddressInfo.COUNTRY, "Canada");
```
3. Create a `Coordinates` object that the location-based services locator server uses as a starting location to search for location information for an address.
```
Coordinates co = new Coordinates(45.423488, -75.697929, 0);
```
4. Create a `Locator` object.
```
Locator lo = new Locator();
```
5. Invoke `Locator.geocode(AddressInfo address, Coordinates startCoords)`.
```
Enumeration en = lo.geocode(ai, co);
```

Retrieve location information for an address stored in a String

1. Create a `String` object that contains address information.
```
String Address = "Toronto, Canada";
```
2. Create a `Coordinates` object that the locator server for location-based services uses as a starting location to search for location information for an address.
```
Coordinates co = new Coordinates(45.423488, -75.697929, 0);
```
3. Create a `Locator` object.
```
Locator lo = new Locator();
```
4. Invoke `Locator.geocode(String freeformString, Coordinates startCoords)`.
```
Enumeration en = lo.geocode(sAddress, co);
```

Cancel a request for location information for an address

Invoke `Locator.cancel()`.
```
lo.cancel();
```

Determine why a request for location information was unsuccessful

1. In your code, create a try block for a `LocationException`.
2. After the try block, create a catch block that invokes `LocationException.getErrorCode()`.
```
try {
} catch (LocationException e) {
    System.err.println("LocationException: "
                       + e.getErrorCode());
}
```

Open BlackBerry Maps from the browser

A BlackBerry® device user can use the browser on a BlackBerry device to access location information on a web server and display that information in BlackBerry Maps. You store location information in .xloc files and place the .xloc files on a web server. When a BlackBerry device user accesses the link for the .xloc file, for example http://servername/Filename.xloc, the browser opens BlackBerry Maps to display the location information. The browser passes the .xloc file as a `String` to the location document parser.

1. Create a file with the extension .xloc.
2. Place the information that you would normally place in a location document in the .xloc file. The location document information is the same information you assign to the `String` variable location document.
```
"<lbs><location lon='-7938675' lat='4367022' label='Toronto, ON'
description='Toronto' zoom='10'/></lbs>";
```

3. On a web server, place the .xloc file in the folder where you store .html files.
4. Register the following MIME type with the web server: text/vnd.rim.location xloc.

Clearing data from the map

You can use the `id` and `clear` attributes of the `<lbs>` element to remove data from a map. When you want BlackBerry Maps to no longer display certain location data, in the next location document you send to BlackBerry Maps, configure the `clear` attribute to the value of the `id` attribute of the location document with the location data to remove.

For example, to display data for Kitchener, Ontario in BlackBerry Maps that you can later remove, give the `id` attribute of the location document a value.

```
String document = "<lbs id='Kit' ><location lon='-0802300' lat='4327000'
label='Kitchener, ON' description='Kitchener' zoom='10'/></lbs>";
```

When you send the location document for Kitchener, Ontario to BlackBerry Maps, BlackBerry Maps displays the information for Kitchener, Ontario.

To remove the information for Kitchener, Ontario from BlackBerry Maps, in the next location document you send to BlackBerry Maps, configure the `clear` attribute to the value of the `id` attribute of the location document that contains information for Kitchener.

```
String document = "<lbs clear='Kit' ><location lon='-7938675' lat='4367022'
label='Toronto, ON' description='Toronto' zoom='10'/></lbs>";
```

When you send the new location document to BlackBerry Maps, BlackBerry Maps removes the Kitchener data from the map.

Clear data from a location document with an id attribute

A location document string with `id` and `clear` attributes uses the following format.

```
<lbs id='<string>'' clear='{NONE | DOCS | LOCATIONS | ROUTES | ALL |
comma_delimited_list_of_document_ids}'>
```

You can only remove data from a location document that has an `id` attribute set to a value.

Create a `String` that configures the `clear` attribute to the `id` of the location document that contains the data to remove from a map.

```
String document = "<lbs clear='Montreal' ><location lon='-7938675'
lat='4367022' label='Toronto, ON' description='Toronto' zoom='10'/></lbs>";
```

Clear data from all location documents with an id attribute

A location document string with `id` and `clear` attributes uses the following format.

```
<lbs id='<string>'' clear='{NONE | DOCS | LOCATIONS | ROUTES | ALL |
comma_delimited_list_of_document_ids}'>
```

You can only remove data from a location document that has an `id` attribute set to a value.

Create a `String` that configures the clear attribute to `Docs`.

```
String document = "<lbs clear='Docs' ><location lon='-7938675' lat='4367022'
label='Toronto, ON' description='Toronto' zoom='10'/></lbs>";
```

Clear data from specific location documents with an id attribute

A location document string with `id` and `clear` attributes uses the following format.

```
<lbs id='<string>'' clear='{NONE | DOCS | LOCATIONS | ROUTES | ALL |
comma_delimited_list_of_document_ids}'>
```

You can only remove data from a location document that has an `id` attribute set to a value.

Create a `String` that configures the `clear` attribute to a comma-delimited list of `id`s.

```
String document = "<lbs clear='MONTREAL','OTTAWA' ><location
lon='-7938675' lat='4367022' label='Toronto, ON' description='Toronto'
zoom='10'/></lbs>";
```

Clear all location data from a map

A location document string with `id` and `clear` attributes uses the following format.

```
<lbs id='<string>'' clear='{NONE | DOCS | LOCATIONS | ROUTES | ALL |
comma_delimited_list_of_document_ids}'>
```

You can only remove data from a location document that has an `id` attribute set to a value.

Create a `String` that configures the `clear` attribute to `Locations`.

```
String document = "<lbs clear='Locations' ><location lon='-7938675'
lat='4367022' label='Toronto, ON' description='Toronto' zoom='10'/></lbs>";
```

Glossary

3

API
application programming interface

GPS
Global Positioning System

JSR
Java® Specification Request

MIME
Multipurpose Internet Mail Extensions

NMEA
National Marine Electronics Association

PDE
Position Determination Entity

SIM
Subscriber Identity Module

WGS
World Geodetic System

XML
Extensible Markup Language

Provide feedback

4

To provide feedback on this deliverable, visit www.blackberry.com/docsfeedback.

Legal notice

5

©2008 Research In Motion Limited. All rights reserved. BlackBerry®, RIM®, Research In Motion®, SureType®, SurePress™ and related trademarks, names, and logos are the property of Research In Motion Limited and are registered and/or used in the U.S. and countries around the world.

Bluetooth is a trademark of Bluetooth SIG. Java is a trademark of Sun Microsystems, Inc. All other trademarks are the property of their respective owners.

The BlackBerry smartphone and other devices and/or associated software are protected by copyright, international treaties, and various patents, including one or more of the following U.S. patents: 6,278,442; 6,271,605; 6,219,694; 6,075,470; 6,073,318; D445,428; D433,460; D416,256. Other patents are registered or pending in the U.S. and in various countries around the world. Visit www.rim.com/patents for a list of RIM (as hereinafter defined) patents.

This documentation including all documentation incorporated by reference herein such as documentation provided or made available at www.blackberry.com/go/docs is provided or made accessible "AS IS" and "AS AVAILABLE" and without condition, endorsement, guarantee, representation, or warranty of any kind by Research In Motion Limited and its affiliated companies ("RIM") and RIM assumes no responsibility for any typographical, technical, or other inaccuracies, errors, or omissions in this documentation. In order to protect RIM proprietary and confidential information and/or trade secrets, this documentation may describe some aspects of RIM technology in generalized terms. RIM reserves the right to periodically change information that is contained in this documentation; however, RIM makes no commitment to provide any such changes, updates, enhancements, or other additions to this documentation to you in a timely manner or at all.

This documentation might contain references to third-party sources of information, hardware or software, products or services including components and content such as content protected by copyright and/or third-party web sites (collectively the "Third Party Products and Services"). RIM does not control, and is not responsible for, any Third Party Products and Services including, without limitation the content, accuracy, copyright compliance, compatibility, performance, trustworthiness, legality, decency, links, or any other aspect of Third Party Products and Services. The inclusion of a reference to Third Party Products and Services in this documentation does not imply endorsement by RIM of the Third Party Products and Services or the third party in any way.

EXCEPT TO THE EXTENT SPECIFICALLY PROHIBITED BY APPLICABLE LAW IN YOUR JURISDICTION, ALL CONDITIONS, ENDORSEMENTS, GUARANTEES, REPRESENTATIONS, OR WARRANTIES OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION, ANY CONDITIONS, ENDORSEMENTS, GUARANTEES, REPRESENTATIONS OR WARRANTIES OF DURABILITY, FITNESS FOR A PARTICULAR PURPOSE OR USE, MERCHANTABILITY, MERCHANTABLE QUALITY, NON-INFRINGEMENT, SATISFACTORY QUALITY, OR TITLE, OR ARISING FROM A STATUTE OR CUSTOM OR A COURSE OF DEALING OR USAGE OF TRADE, OR RELATED TO THE DOCUMENTATION OR ITS USE, OR PERFORMANCE OR NON-PERFORMANCE OF ANY SOFTWARE, HARDWARE, SERVICE, OR ANY THIRD PARTY PRODUCTS AND SERVICES REFERENCED HEREIN, ARE HEREBY EXCLUDED. YOU MAY ALSO HAVE OTHER RIGHTS THAT VARY BY STATE OR PROVINCE. SOME JURISDICTIONS MAY NOT ALLOW THE EXCLUSION OR LIMITATION OF IMPLIED WARRANTIES AND CONDITIONS. TO THE EXTENT PERMITTED BY LAW, ANY IMPLIED WARRANTIES OR CONDITIONS RELATING TO THE DOCUMENTATION TO THE EXTENT THEY CANNOT BE EXCLUDED AS SET OUT ABOVE, BUT CAN BE LIMITED, ARE HEREBY LIMITED TO NINETY (90) DAYS FROM THE DATE YOU FIRST ACQUIRED THE DOCUMENTATION OR THE ITEM THAT IS THE SUBJECT OF THE CLAIM.

TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW IN YOUR JURISDICTION, IN NO EVENT SHALL RIM BE LIABLE FOR ANY TYPE OF DAMAGES RELATED TO THIS DOCUMENTATION OR ITS USE, OR PERFORMANCE OR NON-PERFORMANCE OF ANY SOFTWARE, HARDWARE, SERVICE, OR ANY THIRD PARTY PRODUCTS AND SERVICES REFERENCED HEREIN INCLUDING WITHOUT LIMITATION ANY OF THE FOLLOWING DAMAGES: DIRECT, CONSEQUENTIAL, EXEMPLARY, INCIDENTAL, INDIRECT, SPECIAL, PUNITIVE, OR AGGRAVATED DAMAGES, DAMAGES FOR LOSS OF PROFITS OR REVENUES, FAILURE TO REALIZE ANY EXPECTED SAVINGS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION, LOSS OF BUSINESS OPPORTUNITY, OR CORRUPTION OR LOSS OF DATA, FAILURES TO TRANSMIT OR RECEIVE ANY DATA, PROBLEMS ASSOCIATED WITH ANY APPLICATIONS USED IN CONJUNCTION WITH RIM PRODUCTS OR SERVICES, DOWNTIME COSTS, LOSS OF THE USE OF RIM PRODUCTS OR SERVICES OR ANY PORTION THEREOF OR OF ANY AIRTIME SERVICES, COST OF SUBSTITUTE GOODS, COSTS OF COVER, FACILITIES OR SERVICES, COST OF CAPITAL, OR OTHER SIMILAR PECUNIARY LOSSES, WHETHER OR NOT SUCH DAMAGES WERE FORESEEN OR UNFORESEEN, AND EVEN IF RIM HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW IN YOUR JURISDICTION, RIM SHALL HAVE NO OTHER OBLIGATION, DUTY, OR LIABILITY WHATSOEVER IN CONTRACT, TORT, OR OTHERWISE TO YOU INCLUDING ANY LIABILITY FOR NEGLIGENCE OR STRICT LIABILITY.

THE LIMITATIONS, EXCLUSIONS, AND DISCLAIMERS HEREIN SHALL APPLY: (A) IRRESPECTIVE OF THE NATURE OF THE CAUSE OF ACTION, DEMAND, OR ACTION BY YOU INCLUDING BUT NOT LIMITED TO BREACH OF CONTRACT, NEGLIGENCE, TORT, STRICT LIABILITY OR ANY OTHER LEGAL THEORY AND SHALL SURVIVE A FUNDAMENTAL BREACH OR BREACHES OR THE FAILURE OF THE ESSENTIAL PURPOSE OF THIS AGREEMENT OR OF ANY REMEDY CONTAINED HEREIN; AND (B) TO RIM AND ITS AFFILIATED COMPANIES, THEIR SUCCESSORS, ASSIGNS, AGENTS, SUPPLIERS (INCLUDING AIRTIME SERVICE PROVIDERS), AUTHORIZED RIM DISTRIBUTORS (ALSO INCLUDING AIRTIME SERVICE PROVIDERS) AND THEIR RESPECTIVE DIRECTORS, EMPLOYEES, AND INDEPENDENT CONTRACTORS.

IN ADDITION TO THE LIMITATIONS AND EXCLUSIONS SET OUT ABOVE, IN NO EVENT SHALL ANY DIRECTOR, EMPLOYEE, AGENT, DISTRIBUTOR, SUPPLIER, INDEPENDENT CONTRACTOR OF RIM OR ANY AFFILIATES OF RIM HAVE ANY LIABILITY ARISING FROM OR RELATED TO THE DOCUMENTATION.

Prior to subscribing for, installing, or using any Third Party Products and Services, it is your responsibility to ensure that your airtime service provider has agreed to support all of their features. Some airtime service providers might not offer Internet browsing functionality with a subscription to the BlackBerry® Internet Service. Check with your service provider for availability, roaming arrangements, service plans and features. Installation or use of Third Party Products and Services with RIM's products and services may require one or more patent, trademark, copyright, or other licenses in order to avoid infringement or violation of third party rights. You are solely responsible for determining whether to use Third Party Products and Services and if any third party licenses are required to do so. If required you are responsible for acquiring them. You should not install or use Third Party Products and Services until all necessary licenses have been acquired. Any Third Party Products and Services that are provided with RIM's products and services are provided as a convenience to you and are provided "AS IS" with no express or implied conditions, endorsements, guarantees, representations, or warranties of any kind by RIM and RIM assumes no liability whatsoever, in relation thereto. Your use of Third Party Products and Services shall be governed by and subject to you agreeing to the terms of separate licenses and other agreements applicable thereto with third parties, except to the extent expressly covered by a license or other agreement with RIM.

Certain features outlined in this documentation require a minimum version of BlackBerry® Enterprise Server, BlackBerry® Desktop Software, and/or BlackBerry® Device Software.

The terms of use of any RIM product or service are set out in a separate license or other agreement with RIM applicable thereto. NOTHING IN THIS DOCUMENTATION IS INTENDED TO SUPERSEDE ANY EXPRESS WRITTEN AGREEMENTS OR WARRANTIES PROVIDED BY RIM FOR PORTIONS OF ANY RIM PRODUCT OR SERVICE OTHER THAN THIS DOCUMENTATION.

Research In Motion Limited
295 Phillip Street
Waterloo, ON N2L 3W8
Canada

Research In Motion UK Limited
Centrum House
36 Station Road
Egham, Surrey TW20 9LF
United Kingdom

Published in Canada